ABSTRACT
The ability to use Proc SQL and ODBC to manage data outside of SAS is another feature in SAS’s toolbox that programmers use to accomplish many different tasks. The idea of pulling data from sources outside of SAS is behind both of these tools. A key benefit of using ODBC and to a certain amount Proc SQL is the reduction in coding that is written. Another positive aspect of using Proc SQL and ODBC is the ability to manage data across a network. At National Jewish Medical & Research Center ODBC and Proc SQL are used to manage and pull end users data across networks so users can examine their data quickly and efficiently. Since end users can have almost real time access to the collected data it is especially useful for checking the accuracy of the data from patient health studies and clinical trials. Error checks and data handling are done first as the data is collected and put into data tables with SAS and then outputted either with an ODBC connection or Proc SQL. The purpose of this paper is to show users how to set up an ODBC connection, compare the reduction in code when an ODBC is set up and how to use Proc SQL to manage data outside of SAS.

INTRODUCTION
Open Database Connectivity (ODBC) is an Application Programming Interface (API) that allows a programmer to access data from database management system with a non-native application. ODBC can be used to access data from systems that use Structured Query Language (SQL) as its data access standard. ODBC can also be used to access data in an EXCEL spreadsheet.

When writing code to interact with a database, you usually have to add native code that talks to a particular database using a proprietary language. If you want to access MS-Access, SQL Server or an Oracle database you would need to know and understand three different flavors of SQL with an ODBC in place, you can access the data in these different databases from SAS and use both SQL statements and SAS to manage and access data.

Proc SQL is a SAS procedure that allows a user to actually access, manipulate and change data within the relational database itself, without opening the database inside the SAS system. This can be used when an ODBC has not been set up as a libname. SQL is a language that enables a programmer to create and operate on relational databases, such as MS-Access, MS SQL Server, Oracle and MySQL. A relational database is tables that are related to each other via fields that are shared.

ODBC CONNECTION

ODBC connections are useful in many applications. An ODBC connection can be used with JMP, MS-ACCESS, SAS, SQL-SERVER, Verity Teleforms and in web applications. Under the Windows operating system there are a number of options to use ODBC. These options include USER DSN, SYSTEM DSN, FILE DSN, DRIVERS, TRACING, and CONNECTION POOLING.

The steps to setting up an ODBC connection are as follows for a PC with the Windows XP operating system:

1) Click on the start icon
2) Highlight Settings and then click on Control Panel
3) Double Click on Administrative Tools
4) Double click on Data Sources (ODBC)

This opens to a screen titled ODBC Data Source Administrator. It is here that a user will define the ODBC connection.
If the USER DSN is chosen then the ODBC connections will only be useful on the PC that it is defined. If a user is on a network and wants to make an ODBC connection available to other users than define a connection under the SYSTEM DSN.

CREATING AN ODBC CONNECTION

The

Under either the USER or SYSTEM DSN the method to define a connection is the same. The difference between an USER and SYSTEM DSN is that a USER ODBC is machine specific, while a SYSTEM DSN can be shared across a shared network. When setting up a new connection the steps to are the same for a USER or a SYSTEM DSN. To define the steps are -

A) Click on the ADD button
B) Select the appropriate drive (Example – Microsoft Access Drivers (*.mdb))
C) After Selection, the next window is ODBC Microsoft Access Setup.
D) Fill in the Data Source Name, a short description if desired
E) Select the database that the connection should connect with and click Okay
F) You will be returned to the ODBC Data Source Administrator interface
G) Make sure the new ODBC Connection is on the list of ODBC connects in either the User interface or the System interface. This will ensure that you have an ODBC connection in place.
H) Click Okay and the window closes

USING THE ODBC CONNECTION IN SAS
Using the ODBC connection in SAS is only one line of code. In our example here we have named an ODBC connection only for the user of the machine that created it. The name of the connection is ROSS.

The code is - libname ross odbc dsn="ROSS";

You now can work with data in SAS. There are some restrictions. These restrictions have to deal with permissions that a user has been granted by the system administer or the software’s own internal restrictions. An example of
this is in MS – Access. When a table exists this database SAS cannot replace it. The table needs to be deleted and then replaced.

Creating a table in a database is simple with an ODBC connection.

    Data ross.newtable;
    Set work.oldtable;
    Run;

**PROC SQL**

PROC SQL is SAS’s answer to users who were already using SQL in other applications and wanted to use it in SAS applications and for connections to relational databases. PROC SQL is a powerful tool, that when learned by an user the knowledge transferred to MS-Access, SQL Server or any database that is relational and uses SQL. The five PROC SQL statements that will be shown are –

A) Create table
B) Insert into
C) Alter table
D) Update table
E) Delete

**EXAMPLE**

If an ODBC connection has been made with a relational database, then SAS can perform data manipulations and management upon the tables within it. These can be done through the normal data step code or with PROC SQL. The first SQL statements that will be shown deal with the creation of a table on the database.

Libname ross odbc dsn=’Ross’;
* Establish connection to MS SQL Server data via and ODBC Driver;

PROC SQL;
    CREATE TABLE ross.tbl_lab
        Subject_id  num(4)
        Whitebloodcount  num(10.2)
        Redbloodcount  num(10.2)
        Temp  num(5.2)
        Height  num(5.2)
        Weight  num(5.2)
        Date  num format=Date9. informat=Date9.;
Quit;

The code above creates a table on the Ross database in MS SQL Server. This code can also be used to create a SAS dataset also if the different libname were established. One reason for creating a table on the relational database in this fashion verses using the data step coding is cleaner code.

If you already had a SAS dataset then to create a table on the relational database is very simple.
Libname ross "\\bios_data\shared date\ross";
   * Establish’s libname for SAS datasets.;

PROC SQL;
   CREATE TABLE ross.tbl_lab as
      SELECT * FROM ross1.tbl_lab;
QUIT;

The * is a wildcard for all of the fields in the dataset ross1.tbl_lab.

If there is new data that must be placed into a table, then the inserted code in SQL is used. The code to do this is quite simple and is shown below for one new record –

Libname ross1 "\\bios_data\shared date\ross"
   * Establishes libname for SAS datasets.;
Libname ross odbc dsn='Ross';
   * Establish connection to MS SQL Server data via and ODBC Driver;

PROC SQL;
   INSERT INTO ross.tbl_lab
      SET subject_id      = 123
      SET Whitebloodcount = 103.2
      SET Redbloodcount   = 301.2
      SET Temp            = 99.8
      SET Height          = 168
      SET Weight          = 92
      SET Date            = "08OCT1961"d;
QUIT;

If data from a dataset is placed into an existing table on a database then code is as follows.

PROC SQL:
   INSERT INTO
      ross.tbl_lab(subject_id, whitebloodcount, redbloodcount, temp, height, weight, date)
      SELECT subject_id, whitebloodcount, redbloodcount, temp, height, weight
      FROM ross1.tbl_lab;
QUIT;

In relational database tables there is a primary key in a table if duplicates are not allowed in it. The purpose of a primary key is to insure that there is only one record of this number omitted into the table. When using the INSERT ability of Proc SQL and a record is found to be duplicate the procedure will fail. This is a greater concern when an user tries to insert more than a few records.

Using PROC SQL when an ODBC connection is not present in a SAS program.

When an ODBC connection is not set up as a libname then the connection can be set up in the PROC SQL statement as shown below.

PROC SQL;
* Establish connection to MS SQL Server data via and ODBC Driver;

```
CONNECT TO ODBC(DSN='ROSS' UID='user id' PWD=password);
```

```
CREATE TABLE test AS
SELECT *
FROM CONNECTION TO ODBC
  (select *
   from tbl_labs
   where type = 'V'
  );

QUIT;
```

Below is another method to use when the ODBC connection is not set as a
libname.

```
proc sql;
  connect to sqlservr as ROSS_
  (Server=Bios_SQL
   driver={SQL Server}
   database=Ross_Ped_Asthma_Piolt
   user='Jeff Magouirk'
   password=345def);

  execute (select * into work.test)
  from tbl_labs;

QUIT;
```

CONCLUSION

The use of ODBC connections and PROC SQL makes the work of a programmer easier. ODBC connections let the
power of SAS be used across different databases and even spreadsheets. PROC SQL can reduce the amount of
code one uses and is useful for applications in relational databases.

REFERENCES

Gruber, Martin, Mastering SQL, SYBEX, Alameda, California, 2000

Halpin, Terry, Information Modeling and Relational Databases From Conceptual Analysis To Logical

Open Database Connectivity [HTTP://WWW.ROTH.NET/PERL/ODBC/FAQ/]


CONTACT INFORMATION

(In case a reader wants to get in touch with you, please put your contact information at the end of the paper.)

Your comments and questions are valued and encouraged. Contact the author at:

  Author Name : Jeff Magouirk
  Company : National Jewish Medical & Research Center
  Address : 1400 Jackson Street
  City state ZIP : Denver CO 80206
  Work Phone: 303-398-1046
  Fax:: 303-270-2110
  Email: magouirkj@njc.org
SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.